AMENDMENTS TO THE CLAIMS

This listing of claims replaces all prior versions, and listings, of claims in the application:

Listing of Claims:

1. (Currently Amended) A computer-implemented malware detection system for determining whether an executable script is malware according to~~ its functionality~~ functional variables and subroutines of the executable script, the malware detection system comprising:

a malware signature store including at least one known malware script signature, wherein each malware signature in the malware signature store is a normalized signature of a known malware script; [[and]]

a normalization module that obtains an executable script and generates a normalized signature for the executable script, wherein generating a normalized signature for the executable script comprises normalizing ~~tokens~~ variables and subroutines from the executable script into normalized ~~tokens~~ variables and subroutines conforming to a common format suitable for comparison with that at least one malware signature in the malware signature store, the normalizing comprising renaming variables and subroutines from the executable script according to a common naming convention; and

a comparison module, wherein the comparison module compares the normalized signature of the executable script to the at least one normalized malware signature in the malware signature store;

wherein the malware detection system is configured to:

compare the normalized signature of the executable script to the at least one normalized malware signature in the malware signature store to determine whether the executable script is malware~~; and report whether the executable script is malware according to the determination~~, comprising determining whether the comparison found a complete match between the normalized signature for the executable script and the at least one normalize malware signature, and if so, reporting that the executable script is malware.

2-3. (Cancelled)

4. (Currently Amended) A computer-implemented method for determining whether a computer-executable script is malware according to functional variables and subroutines of the computer-executable script, the method comprising:

using one or more processors to perform the following computer-executable acts:

obtaining an executable script;

generating a first normalized signature for the executable script, wherein the first normalized signature comprises normalized ~~tokens~~ variables and subroutines normalized from corresponding ~~tokens~~ variables and subroutines in the executable script in a common format suitable for comparison to normalized signatures of known malware, and wherein the normalized variables and subroutines comprise variables and subroutines from the executable script that are renamed according to a common naming convention;

comparing the first normalized signature to at least one normalized signature of known malware; and

determining, based on the previous comparison, whether the executable script is malware~~; and reporting the results of the determination as to whether the executable script is malware~~, comprising determining if the first normalized signature for the executable script is a complete match with a normalized signature of known malware, and if so, reporting that the executable script is malware.

5. (Currently Amended) A tangible computer-readable medium bearing computer executable instructions which, when executed on a computing device, carry out a method for determining whether a computer-executable script is malware according to functional variables and subroutines of the computer-executable script, comprising:

obtaining an executable script;

generating a first normalized signature for the executable script, wherein the first normalized signature comprises normalized ~~tokens~~ variables and subroutines normalized from corresponding functional ~~contents of~~ variables and subroutines in the executable script in a common format suitable for comparison to normalized signatures of known malware, and wherein the normalized variables and subroutines comprise variables and subroutines from the executable script that are renamed according to a common naming convention;

comparing the first normalized signature to at least one normalized signature of known malware scripts; and

determining, based on the previous comparison, whether the executable script is malware~~; and reporting the results of the determination as to whether the executable script is malware~~, comprising determining if the first normalized signature for the executable script is a complete match with a normalized signature of known malware, and if so, reporting that the executable script is malware.

6. (Cancelled)

7. (Currently Amended) The malware detection system of Claim [[6]]1, wherein comparing the normalized signature of the executable script to the at lease one normalized malware signature in the malware signature store further ~~configured to~~comprises:

~~if the prior determination indicates that the executable script is~~ determining whether the comparison found a partial match ~~to~~ between the normalized signature for the executable script and the at least one malware signature ~~in the malware signature store~~, and if so:

generate a second normalized signature for the executable script, wherein generating a second normalized signature comprises normalizing ~~tokens~~ variables and subroutines from the executable script into a second common format suitable for comparison with a second normalized malware signature of known malware in the malware signature store; and

determine whether the executable script is malware according to a comparison between the second normalized signature and at least one second normalized signature in the malware signature store.

8. (Currently Amended) The malware detection system of Claim 7, wherein normalizing ~~tokens~~ variables and subroutines from the executable script into a second common format suitable for comparison with a second normalized malware signature of known malware in the malware signature store comprises normalizing ~~tokens~~ variables and subroutines of the executable script into a common name according to each ~~token's~~ variable or subroutine's type.

9. (Currently Amended) The malware detection system of Claim [[6]]1, wherein generating a normalized signature for the executable script further comprises generating a set of normalized ~~tokens~~ subroutines for each subroutine in the executable script.

10-11. (Cancelled)

12. (Currently Amended) The ~~malware detection system~~ method of Claim [[11]]23, wherein normalizing ~~tokens~~ variables and subroutines from the executable script into a second common format suitable for comparison with second normalized malware signatures of known malware in the malware signature store means comprises normalizing ~~tokens~~ variables and subroutines of the executable script into a common name according to each ~~token's~~ variable or subroutine's type.

13. (Cancelled)

14. (Currently Amended) The method of Claim [[13]]4, wherein determining, based on the previous comparison, whether the executable script is malware further comprises:

determining if the first normalized signature for the executable script is a partial match with a normalized signature of known malware, and if so:

generating a second normalized malware signature for the executable script, the second normalized signature comprising ~~tokens~~ variables and subroutines from the executable script normalized into a second common format suitable for comparison with second normalized malware signatures of known malware; and

comparing the second normalized signature for the executable script to second normalized signatures of known malware to determine whether the second normalized signature for the executable script is a complete match to a second normalized signature of known malware, and if so, reporting that the executable script is malware.

15. (Currently Amended) The method of Claim 14, wherein normalizing ~~tokens~~ variables and subroutines from the executable script into a second common format suitable for comparison with second normalized malware signatures of known malware comprises normalizing ~~tokens~~ variables and subroutines of the executable script into a common name according to each ~~token's~~ variable or subroutine's type.

16. (Previously Presented) The method of Claim 14 further comprising comparing the second normalized signature for the executable script to second normalized signatures of known malware to determine whether the second normalized signature for the executable script is a partial match to a second normalized signature of known malware, and if so, reporting that the executable script is potential malware.

17. (Cancelled)

18. (Currently Amended) The computer-readable medium of Claim [[17]]5, wherein determining, based on the previous comparison, whether the executable script is malware further comprises:

determining if the first normalized signature for the executable script is a partial match with a normalized signature of known malware, and if so:

generating a second normalized malware signature for the executable script, the second normalized signature comprising ~~tokens~~ variables and subroutines from the executable script normalized into a second common format suitable for comparison with second normalized malware signatures of known malware; and

comparing the second normalized signature for the executable script to second normalized signatures of known malware to determine whether the second normalized signature for the executable script is a complete match to a second normalized signature of known malware, and if so, reporting that the executable script is malware.

19. (Currently Amended) The computer-readable medium of Claim 18, wherein normalizing variables and subroutines from the executable script into a second common format suitable for comparison with second normalized malware signatures of known malware comprises normalizing variables and subroutines of the executable script into a common name according to each ~~token's~~ variable or subroutine's type.

20. (Previously Presented) The computer-readable medium of Claim 19, wherein the method further comprises comparing the second normalized signature for the executable script to second normalized signatures of known malware to determine whether the second normalized signature for the executable script is a partial match to a second normalized signature of known malware, and if so, reporting that the executable script is potential malware.

21-22. (Cancelled)

23. (New) A computer-implemented method for determining whether a computer-executable script is malware according to functional variables and subroutines of the computer-executable script, the method comprising:

using one or more processors to perform the following computer-executable acts:

obtaining an executable script;

generating a first normalized signature for the executable script, wherein the first normalized signature comprises normalized variables and subroutines normalized from corresponding variables and subroutines in the executable script in a format suitable for comparison to normalized signatures of known malware;

comparing the first normalized signature to at least one normalized signature of known malware; and

determining, based on the previous comparison, whether the executable script is malware, comprising:

determining if the first normalized signature for the executable script is a complete match with a normalized signature of known malware, and if so, reporting that the executable script is malware; and

determining if the first normalized signature for the executable script is a partial match with a normalized signature of known malware, and if so:

generating a second normalized malware signature for the executable script, the second normalized signature comprising variables and subroutines from the executable script normalized into a second common format suitable for comparison with second normalized malware signatures of known malware; and

comparing the second normalized signature for the executable script to second normalized signatures of known malware to determine whether the second normalized signature for the executable script is a complete match to a second normalized signature of known malware, and if so, reporting that the executable script is malware.